VULNERABILITY ASSESSMENT REPORT

SUBJECT: Vulnerability Assessment Report – scanme.nmap.org

From: Azhar Dawood, CyberSecurity Intern

To: Future Interns

Date: 17/02/2026

Target: scanme.nmap.org

Overview

A security assessment was conducted on scanme.nmap.org to identify potential vulnerabilities that could impact business operations, data confidentiality, and system integrity. The assessment used tools like Nmap and OWASP ZAP in a passive scanning configuration

Tools Used

- Nmap - Port scanning and service detection
- OWASP ZAP - Web vulnerability identification

NMap scan configuration

Command used
nmap scanme.nmap.org

Purpose: Identify open ports and services

Duration: 5.78 seconds

OWASP ZAP SCAN CONFIGURATION

Scan Type: Passive Scan only

Target: http://scanme.nmap.org

Purpose: Identify web vulnerabilities without intrusive testing

Nmap assessment

Port 22 (SSH): Remote access point. Risk of unauthorized entry if passwords are weak. Solution: Use key authentication and restrict access.

Port 80 (HTTP): The website. Running outdated Apache version with known vulnerabilities. Solution: Update immediately and add HTTPS.

Port 9929 (nping-echo): Test service. Not needed for business. Solution: Disable if not required.

Port 31337 (Elite): CRITICAL CONCERN - Unusual port associated with hacker terminology. Requires immediate investigation as it may indicate unauthorized access.

Remediation steps

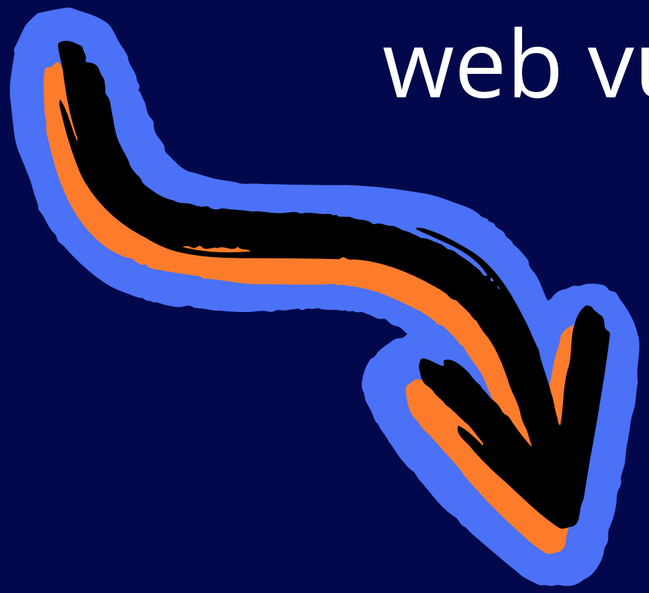
Immediate action(within 24-72 hours)
Investigate port 31337 - identify what service is running,
If unknown and unauthorized, block the port immediately.
Check for signs of server compromise.

Short-term Actions (within 2 weeks)
Update Apache from 2.4.7 to latest version
Implement HTTPS with valid SSL certificate
Harden SSH configuration

Long-term Actions (Monthly)
Regular security patching schedule
Periodic port scans to monitor for unauthorized services
Implement intrusion detection system

OWASP ZAP

web vulnerability scan/identification



The scan revealed 4 medium-risk vulnerabilities related to missing security headers and lack of encryption. The site operates entirely over unencrypted HTTP, exposing all data to potential interception. Multiple information leaks were also found, disclosing server version details that could help attackers target specific vulnerabilities.

FINDING 1: Content Security Policy (CSP) Header Not Set

Without this security rule, if a hacker manages to inject malicious code anywhere on your site, that code can run freely and steal customer information, redirect users to fake login pages, or even take over your website entirely.

Solution

Your IT team needs to add a security rule that tells browsers "only trust content from these specific sources." This means if a hacker tries to load code from their own server, the browser will reject it automatically. Start with a strict rule and then add exceptions only for trusted services like Google Analytics or your own domains.

FINDING 2: HTTP Only Site (No HTTPS)

Your entire website is being sent over an unencrypted connection.

Think of this as sending all your customer information like usernames, passwords and credit card details on postcards instead of sealed envelopes. Anyone along the way, whether someone on the same WiFi at a coffee shop, an internet service provider or a hacker, can read everything.

Solution

Your website needs an SSL certificate and must be configured to use HTTPS. Once installed, all communication between your customers and your website gets encrypted

FINDING 3: Missing Anti-clickjacking Header

Hackers can trick your customers by hiding your website inside an invisible layer on top of their malicious page.

Solution

Your IT team needs to add a simple rule that tells browsers "don't allow this website to be embedded inside other sites as a hidden layer." This prevents malicious sites from wrapping your site in an invisible frame and tricking your users.

FINDING 4: Missing Subresource Integrity (SRI)

Your website loads the Google Analytics script from Google's servers. This is normal and common. But what if Google's servers get hacked? Or what if someone at Google maliciously changes that script? Without integrity checks, any changed version would run on your site automatically, potentially stealing all your customer data.

Solution

When loading external tools like Google Analytics, your developers need to add a digital fingerprint of what the original file should look like. The browser checks this fingerprint every time. If the file has been changed even by one character, the browser blocks it from running.

FINDING 5: In Page Banner Information Leak

Your website is publicly announcing "I'm running Apache version 2.4.7" in the page content. Hackers keep lists of vulnerabilities for every software version, so they now know exactly which weaknesses to try first.

Solution

Your IT team should configure the web server to hide its version number. Instead of announcing "Apache 2.4.7", it should just say "Apache" or nothing at all. This doesn't fix the underlying software, but it makes the hacker's job harder by forcing them to guess your version instead of knowing it.

FINDING 6: Server Leaks Version via "Server" HTTP Response Header

Every time someone visits your site, your web server sends a hidden message in the background that says "Hello, I'm Apache version 2.4.7 running on Ubuntu." This is the digital equivalent of your server wearing a name tag with its exact specifications.

Solution

The IT team needs to turn off these automatic announcements. The server can be configured to share less information. This simple change adds a layer of obscurity that makes targeted attacks slightly harder.

FINDING 7: X-Content-Type-Options Header Missing

Older browsers sometimes try to guess what type of file they're looking at. If you have an image on your site, but a hacker finds a way to hide malicious code inside it, the browser might mistakenly run that code instead of just displaying the picture.

Solution

Add a simple instruction that tells browsers "always trust our file type declarations and don't try to guess." This prevents scenarios where a harmless-looking image could be misinterpreted as executable code

Conclusion

The security assessment of scanme.nmap.org using Nmap and OWASP ZAP identified several vulnerabilities that need attention.

Nmap Findings:

Four open ports were discovered including SSH, HTTP and an unusual port 31337. This port is historically associated with hacker terminology and should be investigated as it increases the attack surface.

ZAP Findings:

Four medium risk issues were identified including missing HTTPS encryption, missing security headers and no integrity checks for external scripts. These could allow attackers to intercept customer data, inject malicious code or trick users. Three low risk information leaks were also found that disclose server version details.